CorrNet3D: Unsupervised End-to-end Learning of Dense Correspondence for 3D Point Clouds

Yiming Zeng[1*]   Yue Qian[1]   Zhiyu Zhu[1]   Junhui Hou[1†]   Hui Yuan[2]   Ying He[3]

[1]City University of Hong Kong   [2]Shandong University   [3]Nanyang Technological University

{ym.zeng, yueqian4-c, zhiyuzhu2-c}@my.cityu.edu.hk, jh.hou@cityu.edu.hk, yhe@ntu.edu.sg

Abstract

Motivated by the intuition that one can transform two aligned point clouds to each other more easily and meaningfully than a misaligned pair, we propose CorrNet3D – the first unsupervised and end-to-end deep learning-based framework – to drive the learning of dense correspondence between 3D shapes by means of deformation-like reconstruction to overcome the need for annotated data. Specifically, CorrNet3D consists of a deep feature embedding module and two novel modules called correspondence indicator and symmetric deformer. Feeding a pair of raw point clouds, our model first learns the pointwise features and passes them into the indicator to generate a learnable correspondence matrix used to permute the input pair. The symmetric deformer, with an additional regularized loss, transforms the two permuted point clouds to each other to drive the unsupervised learning of the correspondence. The extensive experiments on both synthetic and real-world datasets of rigid and non-rigid 3D shapes show our CorrNet3D outperforms state-of-the-art methods to a large extent, including those taking meshes as input. CorrNet3D is a flexible framework in that it can be easily adapted to supervised learning if annotated data are available. The source code and pre-trained model will be available at https://github.com/ZENGYIMING-EAMON/CorrNet3D.git.

1. Introduction

Owing to the flexibility and efficiency in representing 3D objects/scenes as well as the recent advances in 3D sensing technology, 3D point clouds have been widely adopted in various applications, e.g., immersive communication [2], autonomous driving [33], AR/VR [37], etc. Since each camera/scanner produces a point cloud in its own camera space rather than the object space, there is no correspondence between two point clouds (even they represent the same object), which poses great challenges for downstream processing and analysis, such as motion transfer [39], shape editing [24], dynamic point cloud compression [20], object recognition [3], shape retrieval [11], surface reconstruction [7], and many others.

Building dense shape correspondence is a fundamental and challenging problem in computer vision and digital geometry processing. There are a considerable number of methods proposed, which can be roughly classified into two categories: model-based [6, 19, 41, 28] and data-driven [23, 9, 8]. The model-based methods usually use handcrafted features to optimize pre-defined processes. The recent deep learning-based methods train their neural networks in a data-driven manner and improve the performance to a large extent. However, the existing methods either require a large amount of annotated data which are difficult to obtain or assume the connectivity information is available in the input data, i.e., polygonal meshes. This paper focuses on unsupervised learning of dense correspondence between non-rigid 3D shapes in the form of 3D point clouds, but the proposed method can also be used for rigid 3D shapes.

Motivation. Let $\mathbf{A} \in \mathbb{R}^{n \times 3}$ and $\mathbf{B} \in \mathbb{R}^{n \times 3}$ be the two point clouds to be corresponded*, where $\mathbf{a}_i = \{x_i, y_i, z_i\}$, $1 \leq i \leq n$ and $\mathbf{b}_j = \{x'_j, y'_j, z'_j\}$, $1 \leq j \leq n$ are the i-th and j-th 3D points of $\mathbf{A}$ and $\mathbf{B}$, respectively. Fig. 1 illustrates our motivation: if $\mathbf{A}$ and $\mathbf{B}$ are well aligned, it is easier to transform one model to the other. More precisely, denote by $\widehat{\mathbf{A}} \in \mathbb{R}^{n \times 3}$ and $\widehat{\mathbf{B}} \in \mathbb{R}^{n \times 3}$ the re-ordered $\mathbf{A}$ and $\mathbf{B}$ via a designed permutation process, respectively. With a designed reconstruction process, it is expected that we can reconstruct $\mathbf{A}$ (resp. $\mathbf{B}$) from $\widehat{\mathbf{B}}$ (resp. $\widehat{\mathbf{A}}$) more easily and meaningfully than the manner of reconstructing $\mathbf{A}$ (resp. $\mathbf{B}$) from $\mathbf{B}$ (resp. $\mathbf{A}$) directly. Therefore, we can minimize the reconstruction error from $\widehat{\mathbf{B}}$ (resp. $\widehat{\mathbf{A}}$) to $\mathbf{A}$ (resp. $\mathbf{B}$)

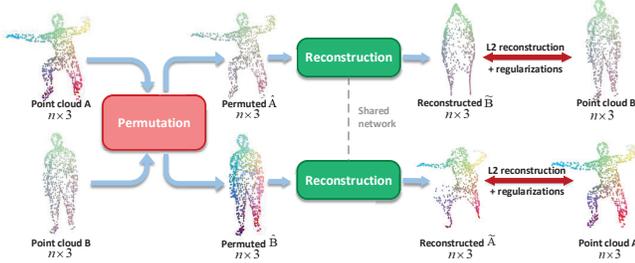*Note that the points are randomly stacked to form a matrix.

Figure 1. Illustration of the motivation of our unsupervised deep learning-based framework for computing dense correspondence between two point clouds.

to drive the learning of the permutation process, which implicitly encodes the dense correspondence between **A** and **B**.

Based on the above intuitive understanding, we propose the *first unsupervised and end-to-end* deep learning-based framework for point clouds. Technically, we propose a novel correspondence indicator and a deformation-like reconstruction module to achieve the permutation and reconstruction processes, respectively. To be specific, the correspondence indicator, fed with point-wise high-dimensional feature representations of the input point clouds learned by a hierarchical feature embedding module, generates a permutation matrix, which explicitly encodes the point-to-point correspondence. During training, the deformation-like reconstruction module receives the aligned point clouds and the global semantic features of inputs to reconstruct each other by optimizing the reconstruction error and additional regularization terms to drive the learning of the permutation matrix.

In summary, we make the following contributions.

1. We propose the first unsupervised deep learning framework for building dense correspondence between point clouds in an end-to-end manner.

2. We propose two novel modules, i.e., the correspondence indicator with the efficient DeSmooth module, the symmetric deformation module, as well as a novel loss function.

3. We show that CorrNet3D can be adapted to both unsupervised and supervised conditions, and handle both non-rigid and rigid shapes well.

4. We experimentally demonstrate the significant superiority of CorrNet3D over state-of-the-art methods. Especially, CorrNet3D even outperforms the method taking 3D meshes as input.

2. Related Work

2.1. Deep Learning for Point Clouds

Unlike well-developed deep convolution neural network (CNN) techniques for 2D images/videos, deep learning

based point cloud processing is more challenging and still in the infant stage, due to its irregular and unorder characteristics. PointNet [31] and PointNet++ [32] are the pioneering works and verify the effectiveness of multi-layer perceptrons (MLPs) in learning point cloud features. DGCNN [48] uses a dynamic graph to aggregate neighborhood information in each layer, and the selection of neighbours is based on feature distances. DCG [45] further boosts DGCNN by encoding additional local connections in coarse-to-fine manner. Volumetric-based methods [49, 26, 34, 46, 22] apply 3D CNNs to process voxelized point clouds; however, they suffer from high computational costs and inevitable quantization errors. In the meantime, inspired by the FoldingNet [50], which learns to deform pre-defined 2D regular grids into 3D shapes, some deformation-based frameworks, such as AtlasNet [14] and 3D-Coded [13], were proposed, which deform a fixed templates (e.g., 2D grid or 3D human mesh) to reconstruct the input point cloud or mesh. Please refer to [16] for the comprehensive survey on deep learning-based point cloud processing.

2.2. Non-rigid Shape Correspondence

Non-rigid shape correspondence or matching aims to find the point-to-point correspondence of two deformable 3D shapes. As an active research area in computer vision and graphics, many methods have been proposed, and one may refer to the surveys [43, 35, 4] for a comprehensive understanding. Here we briefly introduce one stream, i.e., functional map (FM)-based methods [28] which are compared in this paper. Specifically, this kind of methods first performs spectral analysis on 3D meshes to construct an FM, and then optimizes a least-squares problem to convert the resulting FM to point-to-point correspondence under the assumption of non-rigid but isometric deformation. To overcome the deficiency of solving the optimization problem, Litany *et al.* [23] proposed deep FM, which integrates the module of computing FM into a deep neural network. However, both original FM-based methods and deep FM use the handcrafted SHOT descriptors [36] , which may limit their performance. Based on deep FM, Halimi et al. [17] proposed to minimize the surface distortion between meshes to drive the correspondence learning process. Instead of using handcrafted descriptors, the most recent deep learning-based method named deep geometric functional map (DeepGFM) [9] employs KPConv[42] to achieve a data-driven feature representation. Although DeepGFM can achieve state-of-the-art performance, it is only applicable for 3D meshes, and its training has to be supervised by ground-truth FMs, whose construction requires ground-truth correspondence. Moreover, additional post-processing is necessary to obtain the final correspondence.

Recently, FlowNet3D [25] was designed to directly learn scene flows from two consecutive point clouds in the form
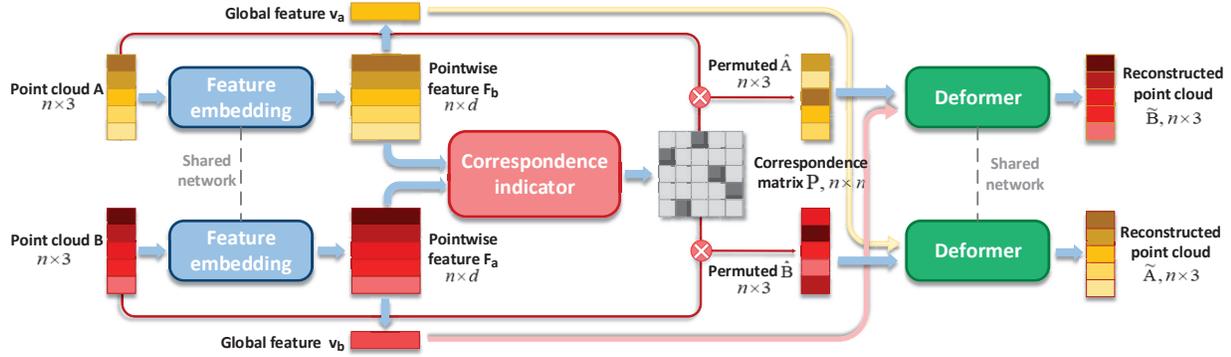
Figure 2. The flowchart of CorrNet3D, an *unsupervised* and *end-to-end* deep learning framework, which aims to obtain a matrix $\mathbf{P}$, which explicitly indicates the correspondence between any two points. We first represent $\mathbf{A}$ and $\mathbf{B}$ with high-dimensional point-wise features $\mathbf{F}_a$ and $\mathbf{F}_b$ as well as the global features $\mathbf{v}_a$ and $\mathbf{v}_b$. Then the correspondence indicator with a novel DeSmooth module takes $\mathbf{F}_a$ and $\mathbf{F}_b$ as input to regress $\mathbf{P}$. To drive the unsupervised learning of $\mathbf{P}$, two symmetric deformers with shared parameters takes the $\mathbf{A}$, $\mathbf{B}$, $\mathbf{P}$ and $\mathbf{v}_a$, and $\mathbf{v}_b$ as inputs to generate the reconstructed point clouds $\widetilde{\mathbf{A}}$ and $\widetilde{\mathbf{B}}$ in the deformation-like manner. CorrNet3D is trained with the reconstruction loss and additional regularization terms on $\mathbf{P}$.

of depth images. To some extent, it can also be used for indicating correspondence, i.e., adding the estimated flow to one point cloud, and then seeking the closest point in the other one. However, such a simple extension may result in serious many-to-one correspondence. Moreover, due to the specific application scenario, FlowNet3D only utilizes the neighborhood information based on the Euclidean distance in two frames, making it not applicable to 3D shapes with serious deformation. Groueix *et al*. [15] proposed a self-supervised approach to achieve deep surface deformation for shapes in the same category, in which the semantic labels from a small set of segmented shapes are transferred to unlabeled data. This work has potential on shape matching.In our experiment, we slightly modified the loss function of FlowNet3D to produce an improved unsupervised model for correspondence prediction, which is adopted as a baseline method for comparisons.

2.3. Rigid Shape Matching

Rigid shape matching or registration aims to obtain a rotation matrix $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ and a translation vector $\mathbf{t} \in \mathbb{R}^{3 \times 1}$ to align two rigid 3D shapes. Over the past decades, a considerable number of methods have been proposed. Please refer to [40] for the comprehensive survey of traditional methods for rigid 3D shape registration. Recently, some deep learning-based methods were proposed. For example, PointNetLK [1] utilizes PointNet [31] to extract global features for two point clouds separately and then estimate $\mathbf{R}$ and $\mathbf{t}$. DCP [47] introduces a transformer [44] to solve the seq-2-seq problem, where the point-wise correspondence and $(\mathbf{R}, \mathbf{t})$ are simultaneously estimated. The recent work RPMNet [51] adopts the Sinkhorn layer [27] to get the correspondence information and weighted SVD [12] to compute $\mathbf{R}$ and $\mathbf{t}$. Note that all these learning-based methods

require ground-truth rotations and translations as supervision or even additional post-processing.

3. Proposed Framework

3.1. Overview

As illustrated in Fig. 2, our CorrNet3D mainly consists of three modules: feature embedding, correspondence indicator, and symmetric deformer. Specifically, we first pass paired input point clouds into the shared feature embedding module to generate point-wise high-dimensional feature embeddings $\mathbf{F}_a \in \mathbb{R}^{n \times d}$ and $\mathbf{F}_b \in \mathbb{R}^{n \times d}$ with d being the feature dimension, which encode their local geometric structures, respectively, and global feature vectors $\mathbf{v}_a \in \mathbb{R}^d$ and $\mathbf{v}_b \in \mathbb{R}^d$, which encode their shape information, respectively. Then we predict a correspondence matrix $\mathbf{P} \in \mathbb{R}^{n \times n}$ by feeding $\mathbf{F}_a$ and $\mathbf{F}_b$ into the correspondence indicator, where the the (i, j)-th element $p_{ij} = 1$ indicates the point $\mathbf{a}_i$ corresponds to $\mathbf{b}_j$. To drive the learning of $\mathbf{P}$ in an unsupervised manner, we propose the symmetric deformer in which we utilize $\mathbf{v}_b$ (resp. $\mathbf{v}_a$) and the permuted point cloud $\widehat{\mathbf{A}}$ (resp. $\widehat{\mathbf{B}}$) to reconstruct $\mathbf{B}$ (resp. $\mathbf{A}$). CorrNet3D is end-to-end trained by directly minimizing $\|\mathbf{A} - \widetilde{\mathbf{A}}\|_F^2 + \|\mathbf{B} - \widetilde{\mathbf{B}}\|_F^2 + \lambda \mathcal{R}(\mathbf{P})$, where $\widetilde{\mathbf{A}} \in \mathbb{R}^{n \times 3}$ and $\widetilde{\mathbf{B}} \in \mathbb{R}^{n \times 3}$ are the reconstructed point clouds, $\| \cdot \|_F$ is the Frobenious norm of a matrix, $\lambda > 0$ is the penalty parameter, and $\mathcal{R}(\mathbf{P})$ stands for the regularization on $\mathbf{P}$.

Remark. The proposed CorrNet3D is fundamentally different from the existing works [51],[9], as the correspondence matrix is driven from the perspective of deformation-like reconstruction, rather than the ground-truth correspondence or the well-known functional maps. In addition, CorrNet3D is able to work as a supervised model by removing the deformation module and employing ground-truth corre-
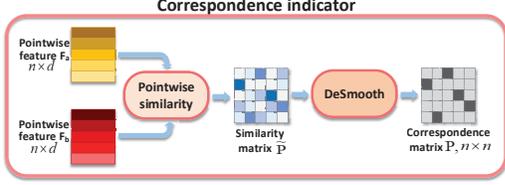
Figure 3. Illustration of the correspondence indicator, which predicts a correspondence matrix $\mathbf{P}$ by taking pointwise features $\mathbf{F}_a$ and $\mathbf{F}_b$ as input.

spondence to supervise the learning of $\mathbf{P}$. In the experiment section, we demonstrate the significant advantage of Corr-Net3D under both unsupervised and supervised scenarios.

3.2. Feature Embedding

We use a shared DNN-based feature learner, namely DGCNN [48], to embed $\mathbf{A}$ and $\mathbf{B}$ to a high-dimensional feature space in a hierarchical manner. Note that other advanced feature representation methods [38], [5] can also be used to further boost performance. To be specific, DGCNN consists of several layers named EdgeConv. For the i-th point, we first calculate the Euclidean distance between features to determine the set of its k nearest neighbours denoted by Ω_i^l. Then, we apply an MLP [18] followed by a max-pooling operator $\square$ to obtain a new feature representation $\mathbf{f}_i^{l+1} = \square_{\mathbf{f}_j^l \in \Omega_i^l} \mathcal{M}_l(\mathbf{f}_i^l, \mathbf{f}_j^l - \mathbf{f}_i^l)$, where $\mathbf{f}_i^l \in \mathbb{R}^{1 \times d}$ be the feature representation of point i fed into the l-th EdgeConv. $\mathbf{f}_i^{l+1}$ is capable of capturing the local geometry structure of point i. After L EdgeConv layers, we can obtain the final point-wise features $\mathbf{F}_a \in \mathbb{R}^{n \times d}$ for $\mathbf{A}$ and $\mathbf{F}_b \in \mathbb{R}^{n \times d}$ for $\mathbf{B}$. By applying another max-avg-pooling operator, the global feature vectors for A and B could be accordingly obtained denoted as $\mathbf{v}_a \in \mathbb{R}^d$ and $\mathbf{v}_b \in \mathbb{R}^d$, respectively. Please refer to [48] for more details about DGCNN.

3.3. Correspondence Indicator

Our correspondence indicator module aims to learn a correspondence matrix $\mathbf{P} \in \mathbb{R}^{n \times n}$ to explicitly indicate the correspondence between any two points of $\mathbf{A}$ and $\mathbf{B}$. Ideally, $\mathbf{P}$ should be a permutation matrix that is binary and orthogonal, and iff $p_{ij} = 1$, point $\mathbf{a}_i$ corresponds to point $\mathbf{b}_j$. However, such a permutation matrix is non-differentiable, making it hard to optimize in a deep learning framework. Alternatively, we regress an approximate doubly stochastic matrix instead, which is differentiable and close to binary. Moreover, there is only a single prominent element dominating each row and column. During inference, we quantize $\mathbf{P}$ to an exact binary matrix $\widehat{\mathbf{P}}$.

As shown in Fig. 3, to learn $\mathbf{P}$, we first measure the similarity between points in the high-dimensional feature

space in the inverse distance sense, i.e.,

$$\widetilde{p}_{ij} = \frac{1}{\|\mathbf{f}_{a,i} - \mathbf{f}_{b,j}\|_2}, \tag{1}$$

where $\widetilde{p}_{ij} \subset \widetilde{\mathbf{P}} \in \mathbb{R}^{n \times n}$, and $\mathbf{f}_{a,i}$ and $\mathbf{f}_{b,j} \in \mathbb{R}^{1 \times d}$ are the i-th and j-th point-wise features corresponding to $\mathbf{a}_i$ and $\mathbf{b}_j$, respectively. However, the resulting $\widetilde{\mathbf{P}}$ is far away from realizing correspondence. To further enhance $\widetilde{\mathbf{P}}$, one can simply adopt Sinkhorn layers [27], which perform the softmax operation on in column-wise and row-wise iteratively and alternatively, as done in [51]; however, the efficiency of Sinkhorn layers is low due to the iterative manner. To tackle this issue, we propose a novel DeSmooth module to improve $\widetilde{\mathbf{P}}$.

DeSmooth Module. Assume that $\widetilde{p}_{ij}$ generally obeys a series of Gaussian distributions[†]: $\widetilde{p}_{ij} \sim \mathcal{N}\left(\mu_i, \sigma_i^2\right)$, where μ_i and σ_i are the mean and standard deviation of the i-th row of $\widetilde{\mathbf{P}}$. We first normalize $\widetilde{p}_{ij}$ in row-wise, i.e., $z_{ij} = \frac{\widetilde{p}_{ij} - \mu_i}{\sigma_i}$. Accordingly, z_{ij} follows a standard normal distribution $z_{ij} = \frac{\widetilde{p}_{ij} - \mu_i}{\sigma_i} \sim \mathcal{N}(0,1)$. Give a prior ratio t to z_{ij}, we have

$$\widetilde{z}_{ij} = t \cdot z_{ij} \sim \mathcal{N}(0, t). \tag{2}$$

For i-th row of $\widetilde{\mathbf{Z}} \in \mathbb{R}^{n \times n}$ ($\widetilde{z}_{ij} \subset \widetilde{\mathbf{Z}}$), we compute the number of elements whose values are not less than a threshold τ, i.e.,

$$c_i = \#\{\widetilde{z}_{ij} | \widetilde{z}_{ij} \geqslant \tau, j = 1, \cdots, n\} \tag{3}$$

where $\#\{\cdot\}$ denotes the cardinality. We expect the value of c_i to be close to 1, which means that in each row there's a high probability that only a single element dominates the row.

The n rows of $\widetilde{\mathbf{Z}}$ could be thought of as n i.i.d events, and thus the set $\mathbf{c} = \{c_i | i = 1, \cdots, n\}$ also follows a Gaussian distribution with the expectation μ_c and variance σ_c depending on the prior ratio t. Therefore, According to the three-sigma rule [30], we can set a proper t to control the bound $[\mu_c - 3\sigma_c, \mu_c + 3\sigma_c]$ to be centered around 1, such that the aforementioned expectation on a feasible correspondence matrix can be realized. Finally, we apply the softmax operation on $\widetilde{z}_{ij}$ again and obtain the correspondence matrix $\mathbf{P}$, i.e.,

$$p_{ij} = \frac{e^{\widetilde{z}_{ij}}}{\sum_{j=1}^{n} e^{\widetilde{z}_{ij}}}. \tag{4}$$

The advantage of our DeSmooth over Sinkhorn layers is also experimentally demonstrated in Sec. 4.5.

3.4. Symmetric Deformer

Given ground-truth correspondence between $\mathbf{A}$ and $\mathbf{B}$, the modules for learning $\mathbf{P}$ can be easily trained in a supervised manner. Annotating large amount of data is,

---

[†]As $\widetilde{p}_{ij}$ is non-negative, it does not follow a strict Gaussian distribution. However, we found such an assumption still works well in practice
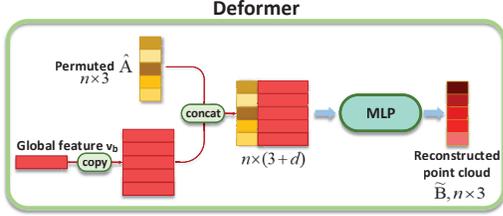
Figure 4. Illustration of one branch of the symmetric deformer. The global feature $\mathbf{v}_b$ encoding the shape information of $\mathbf{B}$ is concatenated to the 3D coordinate of each point of $\widehat{\mathbf{A}}$, which are fed into an MLP to reconstruct $\mathbf{B}$ from a deformation perspective.

however, costly and time-consuming. Inspired by recent deformation-based methods, such as FoldingNet [50] and AtlasNet [14], which feed a pre-defined 2D grid appended with a global feature vector to a network to reconstruct 3D shapes, we propose symmetric deformer shown in Fig. 4, which reconstruct $\mathbf{A}$ and $\mathbf{B}$ in a deformation-like fashion to achieve the *unsupervised* learning of $\mathbf{P}$.

We link the matrix $\mathbf{P}$ with the deformation module based on the aforementioned intuition, i.e., it is more easily and meaningfully to transform two aligned point clouds to each other than a misaligned pair. Specifically, we first permute the input point clouds using the learned $\mathbf{P}$ approximate to a permutation matrix, i.e.,

$$\widehat{\mathbf{A}} = \mathbf{P}^{\mathsf{T}}\mathbf{A}, \quad \widehat{\mathbf{B}} = \mathbf{P}\mathbf{B}. \quad (5)$$

The resulting $\widehat{\mathbf{A}}$ (resp. $\widehat{\mathbf{B}}$) is approximately aligned with $\mathbf{B}$ (resp. $\mathbf{A}$). Then, we deform $\widehat{\mathbf{A}}$ to $\mathbf{B}$ and $\widehat{\mathbf{B}}$ to $\mathbf{A}$, by respectively utilizing their overall shape information encoded in the learned global feature vectors $\mathbf{v}_a$ and $\mathbf{v}_b$. Technically, we concatenate $\mathbf{v}_a$ (resp. $\mathbf{v}_b$) to each point of $\widehat{\mathbf{B}}$ (resp. $\widehat{\mathbf{A}}$) and pass the extended points to a network consisting of MLP layers, leading to reconstructed point clouds $\widetilde{\mathbf{A}}$ (resp. $\widetilde{\mathbf{B}}$). See Sec.4.5 for the experimental validation towards the effectiveness and the deformation behavior of this module.

3.5. Unsupervised Loss Function

To train the proposed CorrNet3D end-to-end, we promote $\widetilde{\mathbf{A}}$ and $\widetilde{\mathbf{B}}$ to be close to $\mathbf{A}$ and $\mathbf{B}$, respectively, which is achieved by

$$\mathcal{L}_{rec}\left(\widetilde{\mathbf{A}}, \widetilde{\mathbf{B}}\right) = \left\|\mathbf{A} - \widetilde{\mathbf{A}}\right\|_F^2 + \left\|\mathbf{B} - \widetilde{\mathbf{B}}\right\|_F^2. \quad (6)$$

Benefiting from the alignment operation involved in CorrNet3D, we are allowed to use such a point-to-point reconstruction loss, which is easier to optimize than the commonly-used CD loss, thus producing better performance. See the ablation study.

In addition to the reconstruction loss, we also propose another two terms to regularize the learning of the correspondence matrix $\mathbf{P}$. The first regularization term is defined

as

$$\mathcal{L}_{perm}(\mathbf{P}) = \left\|\mathbf{P}\mathbf{P}^{\mathsf{T}} - \mathbf{I}_n\right\|_F^2, \quad (7)$$

where $\mathbf{I}$ is the identity matrix of size $n \times n$. Such a term encourages $\mathbf{P}$ to be close to a permutation matrix to eliminate one-to-many correspondence. Second, we utilize the local geometry similarity between the input point cloud and permuted one to promote the learning of $\mathbf{P}$, i.e., neighbouring points in $\mathbf{A}$ (resp. $\mathbf{B}$) should also be neighbours in $\widehat{\mathbf{B}}$ (resp. $\widehat{\mathbf{A}}$), which is mathematically expressed as

$$\mathcal{L}_{mfd}(\mathbf{P}) = \quad (8)$$

$$\sum_{i=1}^{n}\left(\sum_{k \in \Omega_i^a}\frac{\|\mathbf{p}_i\mathbf{B} - \mathbf{p}_k\mathbf{B}\|_2^2}{\|\mathbf{a}_i - \mathbf{a}_k\|_2^2} + \sum_{s \in \Omega_i^b}\frac{\|\mathbf{p}^i\mathbf{A} - \mathbf{p}^s\mathbf{A}\|_2^2}{\|\mathbf{b}_i - \mathbf{b}_s\|_2^2}\right),$$

where $\mathbf{p}_i$ (resp. $\mathbf{p}^i$) is the i-th row (resp. column) of $\mathbf{P}$, and Ω_i^a (resp. Ω_i^b) is the index set of k nearest neighbours of point $\mathbf{a}_i$ (resp. $\mathbf{b}^i$).

Finally, the overall loss function for training CorrNet3D is written as

$$\mathcal{L}\left(\widetilde{\mathbf{A}}, \widetilde{\mathbf{B}}, \mathbf{P}\right) = \quad (9)$$

$$\mathcal{L}_{rec}\left(\widetilde{\mathbf{A}}, \widetilde{\mathbf{B}}\right) + \lambda_1\mathcal{L}_{perm}\left(\mathbf{P}\right) + \lambda_2\mathcal{L}_{mfd}\left(\mathbf{P}\right),$$

where λ_1 and $\lambda_2 > 0$ are the parameters to balance the three terms. See Sec. 4.5 for the experimental validation towards such an unsupervised loss function.

3.6. Pseudo Clustering for Large-scale Point Clouds

As the size of predicted $\mathbf{P}$ depends on that of the input point cloud, directly inferring the correspondence of large-scale point clouds may cause a memory issue. To this end, we propose pseudo clustering, a simple yet effective approach. Specifically, during inference, we first apply a typical sampling method such as farthest point sampling (FPS) on input point clouds to sample a fewer number of points called key points, which are thought of as cluster centers and fed into CorrNet3D, leading to the correspondence of the key points. Then the nearest neighboring points of each key point are found and sorted according to their Euclidean distances to center, and the correspondence of the neighbouring points of two corresponded key points are finally determined if two neighbouring points have the same rank in their own cluster. Such a pseudo clustering enables us to easily apply CorrNet3D to large-scale point clouds.

It is also worth pointing out that such a simple strategy would degrade the performance of our method when directly applied on large-scale point clouds under the condition with sufficient memory to some extent; however, the experiment shows CorrNet3D can still predict more accurate correspondence than the method even trained with 3D meshes, demonstrating the strong ability of our CorrNet3D.

4. Experiments

In this section, we conducted extensive experiments and comparisons on real scanned non-rigid shapes and synthetic non-rigid and rigid shapes to demonstrate superiority of CorrNet3D in both supervised[‡] and unsupervised scenarios.

4.1. Experiment Setting

Datasets. For non-rigid shape correspondence, we adopted Surreal [13] as the training dataset, consisting of 230K samples, which were randomly grouped into 115K training pairs. We conducted the test on the SHREC dataset [9], which has 430 pairs of non-rigid shapes. For rigid shape correspondence, we adopted the training and test dataset splits of the Surreal dataset [13], which contain 230K and 200 samples, respectively, and we randomly rotated and translated the samples to generate 230K pairs and 200 pairs for training and testing, respectively. Note that we chose these datasets in order to keep the same settings as the compared methods, including DeepGFM [9], DCP [47] and RPMNet [51], for fair comparisons. For all the above training data, each point cloud contains 1024 points. The Surreal and SHREC datasets are both synthetic 3D meshes and we randomly picked 1024 vertices to form the point clouds.

Metrics. To fairly and quantitatively compare different methods, we define the corresponding percentage (Corr (%)) to measure correspondence accuracy, i.e.,

$$\text{Corr} = \frac{1}{n} \left\| \widehat{\mathbf{P}} \odot \mathbf{P}_{gt} \right\|_1, \quad (10)$$

where $\odot$ is the Hadamard product of matrices, $\| \cdot \|_1$ is the ℓ_1 norm of a matrix, and $\mathbf{P}_{gt}$ encodes the ground-truth correspondence. Moreover, for a comprehensive comparison, we computed the corresponding percentage of different methods under various tolerant errors defined as $r/dist_{max}$, where $dist_{max} := \max\{\|\mathbf{a}_i - \mathbf{a}_j\|_2, \forall i, j\}$, and r stands for the tolerant radius. It is worth pointing out that the above quantitative evaluation criteria are similar to those used for 3D mesh-based shape correspondence [28, 23] which adopt the geodesic distance as the tolerant error requiring connectivity information. But such information is not available for point clouds, we directly compute the Euclidean distance.

Implementation details. For the parameters in Eq. (9), we empirically set $\lambda_1 = 0.1$ and $\lambda_2 = 0.01$. The shared symmetric deformer consists of a 3-layer MLP. We implemented it with the PyTorch framework [29] on GeForce RTX 2080Ti.We trained the models with Adam [21] optimizer with the learning rate equal to 1e-4 and the batch size equal to 10 for 300 epochs.
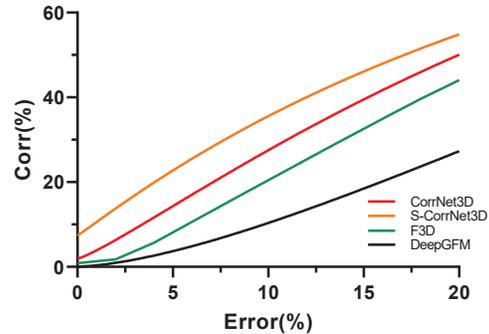

Figure 5. Quantitative comparisons of different methods for non-rigid shape correspondence.
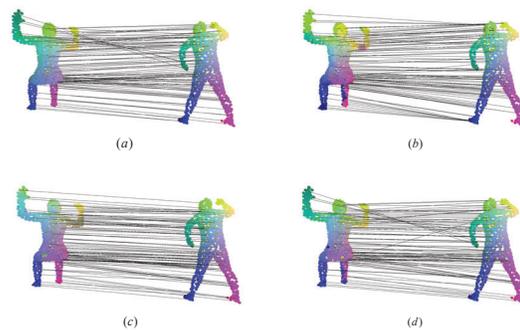

Figure 6. Visual comparisons of different methods for non-rigid shape correspondence in the form of point clouds. Both lines and colors are used for illustrating the correspondence. (a) Ground-truth. (b) F3D. (c) CorrNet3D. (d) S-CorrNet3D.

4.2. Evaluation on Non-rigid Shapes

In this scenario, we compared CorrNet3D and supervised CorrNet3D (S-CorrNet3D) with unsupervised F3D[§] [25] and supervised DeepGFM [9] taking 3D meshes as input.

Fig. 5 shows the quantitative comparisons of different methods, where it can be observed that S-CorrNet3D always produces the best performance, and both CorrNet3D and S-CorrNet3D consistently outperform F3D and Deep-GFM. Especially, the performance advantage of our methods over DeepGFM is more obvious with the tolerant error increasing. Fig. 6 shows visual comparisons of F3D, CorrNet3D and S-CorrNet3D on point clouds, where the predicted correspondence is visualized with colors and lines. From Fig. 6, we can see that CorrNet3D and S-CorrNet3D produce more accurate correspondence than F3D, espe-

---

[‡]The supervised CorrNet3D (S-CorrNet3D) is achieved by removing the symmtric deformation module and training the remaining modules via minimizing the Euclidean distance between the predicted correspondence (i.e., $\mathbf{P}$) and the ground-truth one.

[§]We obtained the unsupervised F3D by modifying the supervised FlowNet3D [25], i.e., we only replaced the loss function with the Chamfer Distance.
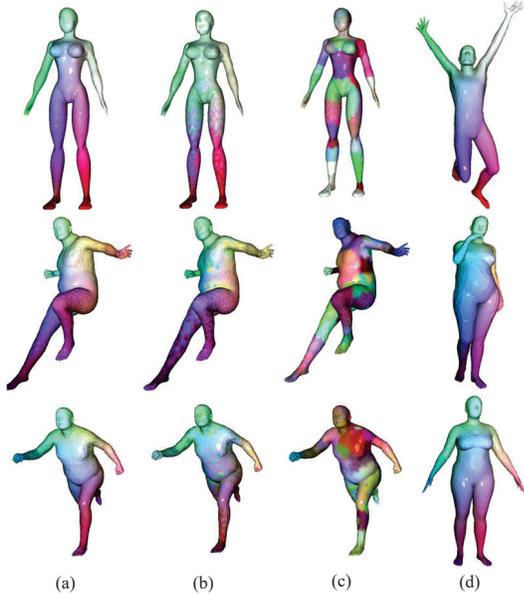
Figure 7. Visual comparisons of different methods on non-rigid shapes in the form of 3D meshes. Each mesh contains 5200 vertices. We render the two corresponded points of the two shapes in the same color. (a) and (d) show the ground-truth correspondence. (b) and (c) show the results of CorrNet3D and DeepGFM, respectively. Note that CorrNet3D takes only 1024 vertices as input and DeepGFM takes 3D meshes as input. We compute the correspondence of the remaining points is obtained via a pseudo clustering strategy.
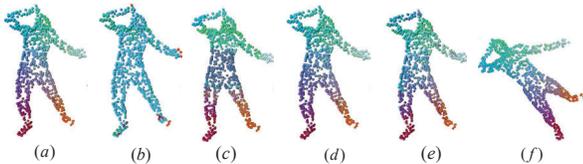

Figure 8. Visual comparisons of different methods for rigid shape correspondence. (a) and (f) show the ground-truth correspondence. (b),(c),(d) and (e) show the results of DCP, RPMNet, CorrNet3D and S-CorrNet3D, respectively.

cially at feet, hands and the right leg. Fig. 7 shows the visual comparisons of CorrNet3D and DeepGFM on 3D meshes, which further demonstrates our method's advantage. That is, the predicted correspondence by our CorrNet3D is closer to the ground-truth one. However, DeepGFM results in patchy distributed wrong correspondence, although it utilizes additional connectivity information and ground-truth correspondence as supervision.

4.3. Evaluation on Rigid Shapes

In this scenario, we compared our CorrNet3D and S-CorrNet3D with DCP [47] and RPMNet [51]. Notice both DCP and RPMNet require ground-truth rigid transforma-
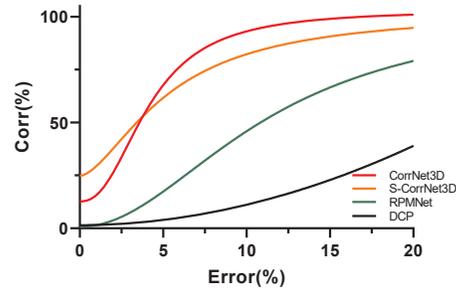

Figure 9. Quantitative comparisons of different methods for rigid shape correspondence.

tion as supervision during training[¶].

Fig. 9 reports the quantitative comparisons of different methods, where it can be seen that CorrNet3D and S-CorrNet3D consistently outperform DCP an RPMNet. Interestingly but not surprisingly, the unsupervised CorrNet3D performs even better than the other three supervised methods. The reason is that the freedom and searching space for the model in the supervised manner will be limited by the training dataset, making it harder to adapt the trained model to data with large transformation. Fig. 8 visually compares the results of different methods, where it can be observed that DCP even fails to obtain correct matching, and RPMNet cannot predict correct matching for hands and the body part. In contrast, S-CorrNet3D and CorrNet3D are able to generate more accurate matching results, which are closer to ground-truth ones.

4.4. Evaluation on Real Scanned Data

We also examined the robustness of CorrNet3D on a real scanned dataset, i.e., 8iVFB [10], including the dynamic point cloud sequences of human motion with serious deformation. The test point clouds contain 1024 points each randomly picked from the original ones. As illustrated in Fig. 10, where the two corresponding points of two shapes predicted by CorrNet3D are visualized with the same color, we can observe CorrNet3D trained on the synthetic dataset still produces impressive performance on real data even with serious deformation, demonstrating the CorrNet3D's strong ability.

4.5. Ablation Study

In this section, we conducted extensive ablation studies for a comprehensive understanding towards our CorrNet3D. We carried out experiments on non-rigid Surreal dataset in the unsupervised scenario.

[¶]As DCP and RPMNet are able to generate soft correspondence by transformer and Sinkhorn, respectively, We output the soft correspondence matrix and set each row's max value as 1, and others as 0 to get a binary correspondence matrix for comparison.

Figure 10. Visual results of CorrNet3D on 8iVFB [10] - a real scanned dataset.
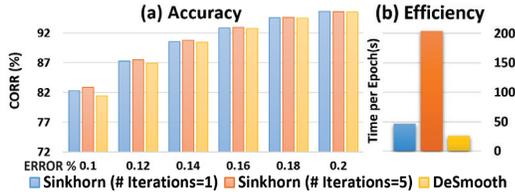

Figure 11. Comparisons between our DeSmooth and the Sinkhorn layer in terms of accuracy (a) and efficiency (b).

Table 1. Validation of the effectiveness and behavior of the symmetric deformer. Here the tolerance error is equal to 20%.

Module	(a) Fully connected	(b) Deformer (w/o $\mathbf{v}_a$, $\mathbf{v}_b$)	(c) Deformer (not shared)	(d) Deformer (shared)
Corr(%)	26.21	25.24	95.97	95.61

DeSmooth module. We compared our DeSmooth with the Sinkhorn layer. Specifically, we replaced the DeSmooth module of CorrNet3D with the Sinkhorn layer, while keep all the remaining settings the same. As shown in Fig. 11, it can be observed that under the same tolerant error, the accuracy of our DeSmooth is on par with that of the Sinkhorn layer, while our DeSmooth improves the efficiency up to 8×.

Deformation module. The results listed in Table 1 validate the effectiveness of our symmetric deformer design and its deformation-like behavior. That is, The correspondence accuracy of the other two settings, i.e., (a) replacing the symmetric deformer of CorrNet3D with fully connected (FC) layers and (b) training CorrNet3D without (w/o) feeding global features into the symmetric deformer (i.e., the absence of the shape information), decreases significantly. We also compared the symmetric deformer w/ and w/o shared parameters As listed in Table 1, comparable performance is achieved under these two settings, but the deformer w/ shared parameters is more memory-efficient as the number of parameters reduces by half.

Loss function. In Table 2, we compared the performance of CorrNet3D when trained with different loss functions. The effectiveness of the regularization terms can be validated by comparing the 2nd and 4th columns, and the advantage of our Euclidean distance-based reconstruction loss over the Chamfer Distance can be demonstrated by comparing the 3rd and 4th columns.

Failure cases. Here we present two failure cases, which occur for computing non-rigid shape correspondence on

Table 2. Validation of the effectiveness of loss setting in Eq. (9), $\mathcal{L}_{reg} = \lambda_1 \mathcal{L}_{perm} + \lambda_2 \mathcal{L}_{mfd}$. The tolerance error is 20%.

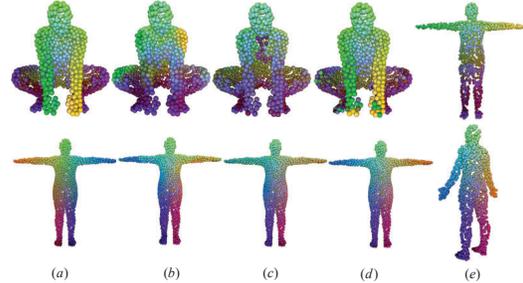| Loss	$\mathcal{L}_{rec}$	$\mathcal{CD} + \mathcal{L}_{reg}$	$\mathcal{L}_{rec} + \mathcal{L}_{reg}$
Corr(%)	31.49	25.74	95.61

Figure 12. Visualization of two failure cases of non-rigid shape correspondence. (a) and (e) show the ground-truth correspondence. (b) and (e) show the results by F3D. (c) and (e) show the results by CorrNet3D. (d) and (e) show the results by S-CorrNet3D. The corresponding points are shown as the same color. Note that the two unsupervised methods F3D and CorrNet3D fail, while the supervised S-CorrNet3D can obtain correct results.

highly symmetric and distorted shapes. As show in Fig. 12, we can see that the two unsupervised methods i.e., F3D and CorrNet3D, both generate wrong correspondence especially in the hands and feet. However, the supervised S-CorrNet3D can successfully obtain the correct correspondence. In future, solving such issues without using heavy annotations or simple data augmentation is a promising direction.

5. Conclusions

We have presented the first unsupervised and end-to-end learning framework named CorrNet3D for building dense correspondence between 3D shapes in the form of point clouds. Unlike existing works, we addressed this challenging problem from the deformation-like reconstruction perspective. Note that CorrNet3D is a flexible framework in that it can be simplified to work in a supervised manner when annotated data are available. We demonstrated the significant advantages of our methods over state-of-the-art ones by conducting extensive experiments on real scanned and synthetic data including rigid and non-rigid shapes in both unsupervised (CorrNet3D) and supervised (S-CorrNet3d) scenarios, as well as comprehensive ablation studies. We believe our methods will bring benefits to other tasks, such as point cloud sequence compression which needs correspondence for eliminating the inter-frame redundancy, and deep learning-based point cloud sequence analysis, which usually has to align points from different frames for feature aggregation.

References

[1] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7163–7172, 2019. 3

[2] John G Apostolopoulos, Philip A Chou, Bruce Culbertson, Ton Kalker, Mitchell D Trott, and Susie Wee. The road to immersive communication. *Proceedings of the IEEE*, 100(4):974–990, 2012. 1

[3] Eduardo Arnold, Omar Y Al-Jarrah, Mehrdad Dianati, Saber Fallah, David Oxtoby, and Alex Mouzakitis. A survey on 3d object detection methods for autonomous driving applications. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3782–3795, 2019. 1

[4] Silvia Biasotti, Andrea Cerri, Alex Bronstein, and Michael Bronstein. Recent trends, applications, and perspectives in 3d shape similarity assessment. In *Computer Graphics Forum*, volume 35, pages 87–119. Wiley Online Library, 2016. 2

[5] Alexandre Boulch. Convpoint: Continuous convolutions for point cloud processing. *Computers & Graphics*, 88:24–34, 2020. 4

[6] Alexander M Bronstein, Michael M Bronstein, and Ron Kimmel. Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences*, 103(5):1168–1172, 2006. 1

[7] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. 1

[8] Étienne Corman, Maks Ovsjanikov, and Antonin Chambolle. Supervised descriptor learning for non-rigid shape matching. In *European Conference on Computer Vision*, pages 283–298. Springer, 2014. 1

[9] Nicolas Donati, Abhishek Sharma, and Maks Ovsjanikov. Deep geometric functional maps: Robust feature learning for shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8592–8601, 2020. 1, 2, 3, 6

[10] Eugene d'Eon, Bob Harrison, Taos Myers, and Philip A Chou. 8i voxelized full bodies-a voxelized point cloud dataset. *ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document WG11M40059/WG1M74006*, 2017. 7, 8

[11] Abubakar Sulaiman Gezawa, Yan Zhang, Qicong Wang, and Lei Yunqi. A review on deep learning approaches for 3d data representations in retrieval and classifications. *IEEE Access*, 8:57566–57593, 2020. 1

[12] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. In *Linear Algebra*, pages 134–151. Springer, 1971. 3

[13] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 3d-coded: 3d correspondences by deep deformation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 230–246, 2018. 2, 6

[14] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018. 2, 5

[15] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. Unsupervised cycle-consistent deformation for shape matching. In *Computer Graphics Forum*, volume 38, pages 123–133. Wiley Online Library, 2019. 3

[16] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020. 2

[17] Oshri Halimi, Or Litany, Emanuele Rodola, Alex M Bronstein, and Ron Kimmel. Unsupervised learning of dense shape correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4370–4379, 2019. 2

[18] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009. 4

[19] Qi-Xing Huang, Bart Adams, Martin Wicke, and Leonidas J Guibas. Non-rigid registration under isometric deformations. In *Computer Graphics Forum*, volume 27, pages 1449–1457. Wiley Online Library, 2008. 1

[20] Tianxin Huang and Yong Liu. 3d point cloud geometry compression on deep learning. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 890–898, 2019. 1

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[22] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 863–872, 2017. 2

[23] Or Litany, Tal Remez, Emanuele Rodola, Alex Bronstein, and Michael Bronstein. Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5659–5667, 2017. 1, 2, 6

[24] Weiping Liu, Jia Sun, Wanyi Li, Ting Hu, and Peng Wang. Deep learning on point clouds and its application: A survey. *Sensors*, 19(19):4188, 2019. 1

[25] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 529–537, 2019. 2, 6

[26] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928, 2015. 2

[27] Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning latent permutations with gumbel-sinkhorn networks. *arXiv preprint arXiv:1802.08665*, 2018. 3, 4

[28] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (TOG)*, 31(4):1–11, 2012. 1, 2, 6

[29] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 6

[30] Friedrich Pukelsheim. The three sigma rule. *The American Statistician*, 48(2):88–91, 1994. 4

[31] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2, 3

[32] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. 2

[33] Tirumalapudi Raviteja. An introduction of autonomous vehicles and a brief survey. *Journal of Critical Reviews*, 7(13):196–202, 2020. 1

[34] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017. 2

[35] Yusuf Sahillioğlu. Recent advances in shape correspondence. *The Visual Computer*, 36(8):1705–1721, 2020. 2

[36] Samuele Salti, Federico Tombari, and Luigi Di Stefano. Shot: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125:251–264, 2014. 2

[37] Rodrigo Silva, Jauvane C Oliveira, and Gilson A Giraldi. Introduction to augmented reality. *National laboratory for scientific computation*, 11, 2003. 1

[38] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua Siegel, and Sanjay Sarma. Pointgrow: Autoregressively learned point cloud generation with self-attention. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 61–70, 2020. 4

[39] Yang-Tian Sun, Qian-Cheng Fu, Yue-Ren Jiang, Zitao Liu, Yu-Kun Lai, Hongbo Fu, and Lin Gao. Human motion transfer with 3d constraints and detail enhancement. *arXiv preprint arXiv:2003.13510*, 2020. 1

[40] G. K. L. Tam, Z. Cheng, Y. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X. Sun, and P. L. Rosin. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *IEEE Transactions on Visualization and Computer Graphics*, 19(7):1199–1217, 2013. 3

[41] Art Tevs, Alexander Berner, Michael Wand, Ivo Ihrke, and H-P Seidel. Intrinsic shape matching by planned landmark sampling. In *Computer Graphics Forum*, volume 30, pages 543–552. Wiley Online Library, 2011. 1

[42] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6411–6420, 2019. 2

[43] Oliver Van Kaick, Hao Zhang, Ghassan Hamarneh, and Daniel Cohen-Or. A survey on shape correspondence. In *Computer Graphics Forum*, volume 30, pages 1681–1707. Wiley Online Library, 2011. 2

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. 3

[45] Kaiqi Wang, Ke Chen, and Kui Jia. Deep cascade generation on point sets. In *IJCAI*, volume 2019, page 4, 2019. 2

[46] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):1–11, 2017. 2

[47] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3523–3532, 2019. 3, 6, 7

[48] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 2019. 2, 4

[49] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 2

[50] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018. 2, 5

[51] Zi Jian Yew and Gim Hee Lee. Rpm-net: Robust point matching using learned features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11824–11833, 2020. 3, 4, 6, 7